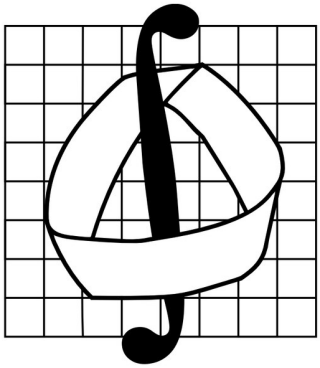
Московский государственный университет имени М. В. Ломоносова
механико-математический факультет
кафедра математической теории интеллектуальных систем



Курсовая работа
студента 431 группы
Мирошниченко Никиты Игоревича

Репараметризация в нейронных сетях

Reparameterization in neural networks

Научный руководитель:
м.н.с., к.ф.-м.н.
Иванов Илья Евгеньевич

Москва, 2025

1 Введение

Важным аспектом современного анализа нейронных сетей является изучение архитектурных изменений, которые позволяют повысить эффективность обучения, снизить количество параметров и улучшить обобщающую способность моделей. Одним из ключевых инструментов в этом направлении является репараметризация — изменение параметрического представления или архитектуры модели, которое улучшает поведение градиентов и оптимизацию на этапе обучения, но не меняет вычисления на этапе инференса.

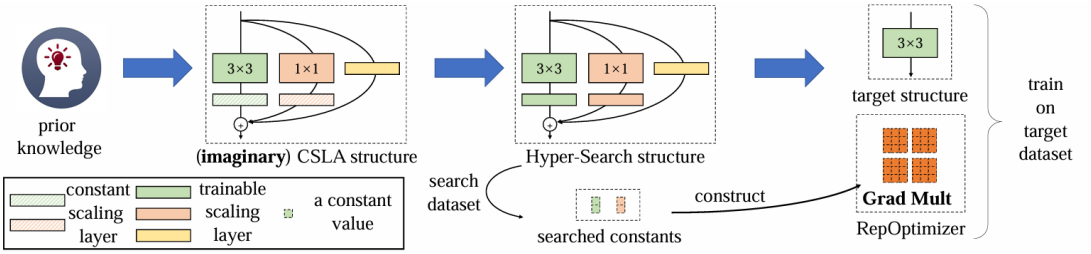


Рис. 1: Схема репараметризации архитектуры нейронной сети в RepOptimizer [3]

Изначально концепция репараметризации получила известность в байесовском машинном обучении (например, в Variational Autoencoders), но в последнее время этот подход всё чаще применяется к архитектурам нейронных сетей. Например, фиктивное усложнение архитектуры на этапе обучения может существенно ускорить сходимость, облегчить распространение градиентов и улучшить качество итоговой модели. После завершения обучения такая сеть может быть преобразована в более компактную форму без потери точности, что особенно важно для задач с ограниченными вычислительными ресурсами.

Примеры из практики. В ResNet репараметризация проявляется через использование *skip connections*, которые изменяют ландшафт оптимизации, улучшая распространение градиентов. Архитектура RepVGG [2] строится по принципу «репараметризуемого» обучения, где во время тренировки используется комбинация свёрточных блоков и batch normalization, а на инференсе эта структура схлопывается в простые последовательные свёртки. Ещё один современный пример — CSLA и RepOptimizer из статьи [3], где архитектура сначала усложняется фиктивными компонентами, затем проводится поиск по гиперпараметрам, а итоговая модель репараметризуется в компактную форму с использованием специальных оптимизаторов.

Целью данной курсовой работы является разработка и анализ математической модели, описывающей влияние репараметризации на процесс обучения нейронной сети. В рамках исследования будут рассмотрены следующие задачи:

- Формализация математической модели, учитывающей разложение весов слоя на матричные множители.
- Анализ градиентных обновлений при обучении репараметризованной модели.
- Исследование влияния параметризации на геометрию пространства оптимизации и динамику сходимости.

Работа устроена следующим образом: в разделе 2 и 3 рассматриваются математические предпосылки появления новой формулы для оптимизатора. Описание модели для сравнения стандартного SGD и с добавочным членом проводится в разделе 4.Результаты сравнения описаны в разделе 5. Выводы и дальнейшие планы содержатся в разделе 6.

2 Матричные разложения

Рассмотрим один полносвязный слой. Пусть на вход подаётся вектор $x \in \mathbb{R}^n$, а на выходе получается вектор $y \in \mathbb{R}^m$. Тогда вычисление слоя можно записать как:

$$y = Ax + b, \tag{1}$$

где $A \in \mathbb{R}^{m \times n}$ — матрица весов слоя, а $b \in \mathbb{R}^m$ — вектор смещения (bias). Обозначим этот слой как FC (Fully Connected).

Предположим теперь, что матрица A может быть разложена в произведение двух матриц:

$$A = W_1 W_2, \tag{2}$$

где $W_1 \in \mathbb{R}^{m \times r}$ и $W_2 \in \mathbb{R}^{r \times n}$.

На основе этого рассмотрим две разные архитектуры:

1. **Обычный полносвязный слой:**

$$y = Ax + b.$$

2. **Два последовательных слоя:**

$$\begin{aligned} h &= W_2 x, \\ y &= W_1 h + b = W_1 W_2 x + b. \end{aligned}$$

Таким образом, на этапе инференса обе структуры полностью эквивалентны:

$$y = Ax + b = W_1 W_2 x + b.$$

Вопрос: что произойдёт на этапе обучения?

Хотя обе схемы дают одинаковый результат на выходе, обучение будет происходить по-разному:

- Параметры W_1 и W_2 обновляются независимо, что приводит к изменению геометрии пространства оптимизации.
- Факторизация может облегчить обучение благодаря более компактному представлению.
- Вариант с двумя слоями может действовать как форма регуляризации или снизить переобучение.
- Градиенты распространяются иначе, что влияет на стабильность и скорость сходимости.

Анализ обучения при разложении матрицы весов

Рассмотрим использование стохастического градиентного спуска (SGD) для обучения весов. Исследуем, как будет меняться произведение $A = W_1 W_2$ при независимом обновлении матриц W_1 и W_2.

Обновление параметров с использованием SGD:

$$W_1^{t+1} = W_1^t - \eta \frac{\partial \mathcal{L}}{\partial W_1}, \tag{3}$$

$$W_2^{t+1} = W_2^t - \eta \frac{\partial \mathcal{L}}{\partial W_2}, \tag{4}$$

где t — момент времени (не степень матрицы, а обычный индекс), η — скорость обучения, $\mathcal{L}$ — функция потерь.

Выражения для градиентов через производную по A:

$$\frac{\partial \mathcal{L}}{\partial W_2} = W_1^\top \frac{\partial \mathcal{L}}{\partial A}, \tag{5}$$

$$\frac{\partial \mathcal{L}}{\partial W_1} = \frac{\partial \mathcal{L}}{\partial A} W_2^\top. \tag{6}$$

Тогда произведение обновлённых матриц даёт:

$$\begin{aligned} W_1^{t+1} W_2^{t+1} &= \left(W_1^t - \eta \frac{\partial \mathcal{L}}{\partial W_1} \right) \left(W_2^t - \eta \frac{\partial \mathcal{L}}{\partial W_2} \right) \\ &= W_1^t W_2^t - \eta \left(\frac{\partial \mathcal{L}}{\partial W_1} W_2^t + W_1^t \frac{\partial \mathcal{L}}{\partial W_2} \right) + \mathcal{O}(\eta^2) \\ &= A^t - \eta \left(\frac{\partial \mathcal{L}}{\partial A} W_2^{t\top} W_2^t + W_1^t W_1^{t\top} \frac{\partial \mathcal{L}}{\partial A} \right) + \mathcal{O}(\eta^2). \end{aligned}$$

Таким образом, обновление произведения W_1W_2 не совпадает с обычным градиентным шагом по A, даже при $A = W_1W_2$. Полученная формула содержит дополнительные матрицы W_1^t, W_2^t, которые "модулируют"градиент. Это приводит к **иной геометрии обучения** и может влиять на траекторию оптимизации и её сходимость.

Условия эквивалентности обновлений

Но при каких условиях формула обновления весов останется той же самой?

Предположим, что матрица A квадратная, а матрицы W_1^t и W_2^t удовлетворяют условиям:

$$\left(W_2^t\right)^\top W_2^t = \alpha E, \tag{7}$$

$$W_1^t \left(W_1^t\right)^\top = \beta E, \tag{8}$$

где $\alpha + \beta = 1$, E - единичная матрица.

Тогда, пренебрегая членом порядка $\mathcal{O}(\eta^2)$, можно утверждать, что:

$$W_1^{t+1}W_2^{t+1} = A^t - \eta\frac{\partial \mathcal{L}}{\partial A} = A^{t+1},$$

т.е. обновление произведения W_1W_2 совпадает с прямым обновлением матрицы A.

Теорема. *Пусть $A^t = W_1^tW_2^t$, и градиентный шаг по W_1 и W_2 выполняется независимо с использованием стохастического градиентного спуска. Тогда, если выполнены следующие условия:*

1. $(W_2^t)^\top W_2^t = \alpha E$,
2. $W_1^t \left(W_1^t\right)^\top = \beta E$,
3. $\alpha + \beta = 1$,

то, с точностью до членов порядка $\mathcal{O}(\eta^2)$, обновление весов эквивалентно обновлению исходной матрицы:

$$W_1^{t+1}W_2^{t+1} = A^{t+1}.$$

Мы уже доказали эту теорему, разложив произведение обновлённых весов и подставив выражения для градиентов.

Но это разложение накладывает слишком строгие ограничения на матрицу A.

Ослабление условий эквивалентности

Попробуем ослабить условия. Пусть $A, W_1, W_2 \in \mathbb{R}^{n\times n}$, и W_1 — ортогональная матрица, то есть:

$$W_1 W_1^\top = W_1^\top W_1 = E.$$

Положим $A = W_1 W_2$. Тогда из ортогональности W_1 получаем:

$$W_1^\top A = W_2.$$

Тогда:

$$W_1^{t+1} W_2^{t+1} = A^t - \eta \left(\frac{\partial \mathcal{L}}{\partial A} \left(A^t\right)^\top W_1^t \left(W_1^t\right)^\top A^t + \frac{\partial \mathcal{L}}{\partial A} \right) + \mathcal{O}(\eta^2),$$

а значит,

$$W_1^{t+1} W_2^{t+1} = A^t - \eta \left(\frac{\partial \mathcal{L}}{\partial A} \left(A^t\right)^\top A^t + \frac{\partial \mathcal{L}}{\partial A} \right).$$

Если обозначить $A^{t+1} := W_1^{t+1} W_2^{t+1}$, то получим новую формулу обновления весов для матрицы A полносвязного слоя.

Заметим, что разложение матрицы A в произведение $W_1 W_2$ фиктивно, то есть на практике мы не раскладываем матрицу A на множители. Но нужно представить хотя бы одно разложение, чтобы эти рассуждения были разумными. Например:

$$A = E \cdot A, \quad \text{т.е. } W_1 = E \text{ — ортогональная матрица}, \quad W_2 = A.$$

3 Случай неквадратной матрицы

Но случай квадратной матрицы встречается крайне редко. Посмотрим, как меняются рассуждения, если

$$A \in \mathbb{R}^{m\times n}, \quad W_1 \in \mathbb{R}^{m\times r}, \quad W_2 \in \mathbb{R}^{r\times n}.$$

Для этого докажем вспомогательный факт.

Замечание Если $A \in \mathbb{R}^{m\times l}$, $m \le l$, и $AA^\top = E_m$, то

$$\mathrm{rk}(A) = m.$$

Доказательство.

$$\mathrm{rk}(AA^\top) = m \quad \text{(по условию)}$$

$$\mathrm{rk}(AA^\top) \le \min(\mathrm{rk}(A), \mathrm{rk}(A^\top)) = \mathrm{rk}(A)$$

Но, с другой стороны,

$$\mathrm{rk}(A) \le \min(m, l) = m$$

Итого:

$$m \leq \mathrm{rk}(A) \leq m,$$

откуда следует, что $\mathrm{rk}(A) = m$. □

Пусть $m > n$. Тогда рассмотрим два случая:

1. Если $r < m$ и $W_1 W_1^\top = E_m$, то $\mathrm{rk}(W_1) \geq m$, что противоречит тому, что $\mathrm{rk}(W_1) \leq \min(m, r) = r$.

2. Поэтому $r \geq m$.

Тогда из условий $W_1 W_1^\top = E_m$ и $A = W_1 W_2$ не получится получить информацию о $W_2^\top W_2$.

Поэтому наложим ещё одно условие:

$$W_1^\top W_1 = E_r,$$

но тогда $\mathrm{rk}(W_1) = m = r$, что означает, что матрица W_1 является квадратной и ортогональной.

Тогда из условия $A = W_1 W_2$ следует, что

$$W_1^\top A = E_r W_2 = W_2.$$

Теперь применим полученные формулы

Вспомним формулу обновления произведения матриц:

$$W_1^{t+1} W_2^{t+1} = A^t - \eta \left(\frac{\partial \mathcal{L}}{\partial A} \left(W_2^t\right)^\top W_2^t + W_1^t \left(W_1^t\right)^\top \frac{\partial \mathcal{L}}{\partial A} \right)$$

$$= A^t - \eta \left(\frac{\partial \mathcal{L}}{\partial A} \left(A^t\right)^\top E_m A^t + E_m \frac{\partial \mathcal{L}}{\partial A} \right)$$

$$= A^t - \eta \left(\frac{\partial \mathcal{L}}{\partial A} \left(A^t\right)^\top A^t + \frac{\partial \mathcal{L}}{\partial A} \right)$$

Получили то же самое.

Осталось предъявить разложение. Например, разложение SVD:

$$A = S \cdot V \cdot D, \quad \text{где} \quad S \in \mathbb{R}^{m \times m} \text{ — ортогональная матрица,}$$

матрица

$$V \in \mathbb{R}^{m \times n}$$

на главной диагонали имеет сингулярные числа, а вне главной диагонали нули.

$$D \in \mathbb{R}^{n \times n}$$

— ортогональная.

Тогда, обозначив $W_1 = S, W_2 = V \cdot D$, получим нужное нам разложение.

Таким образом, мы получили новую форму обновления весов матрицы любого полносвязного слоя.

4 Проверка нового оптимизатора

В данном разделе представлена модель, предназначенная для проверки кастомной формулы обновления весов на автоэнкодере. Модель состоит из энкодера и декодера, но с учётом использования кастомного оптимизатора. На примере этой модели выясним, имеет ли новый оптимизатор преимущества по сравнению с обычным SGD.

4.1 Архитектура автоэнкодера

Модель автоэнкодера включает в себя следующие компоненты:

- **Энкодер:**
  - Входной слой: преобразует входные изображения размером 28×28 в одномерный вектор размера 784.
  - Линейный слой: уменьшает размерность до 128.
  - Функция активации: ReLU.
  - Линейный слой: преобразует в скрытое представление размером $latent\_dim$.
- **Декодер:**
  - Линейный слой: увеличивает размерность до 128.
  - Функция активации: ReLU.
  - Линейный слой: восстанавливает размерность до 784.
  - Функция активации: Sigmoid.
  - Выходной слой: преобразует данные обратно в формат $(1, 28, 28)$.

4.2 Обучение модели

Обучение модели происходит на наборе данных **Fashion-MNIST**. Модель обучается с использованием нового оптимизатора, который применяет обновление весов по формуле:

$$A_{t+1} = A_t - \eta \left(\frac{\partial A}{\partial L} + \frac{\partial A}{\partial L} A_t^\top A_t \right),$$

где A_t — матрица весов на итерации t, η — скорость обучения, а $\frac{\partial L}{\partial A}$ — градиент функции потерь по параметрам A.

4.3 Гиперпараметры

Для обучения модели использовались следующие гиперпараметры:

- Размерность латентного пространства: $latent\_dim = 16$.
- Размер батча: $batch\_size = 128$.
- Количество эпох: $epochs = 100$.
- Начальная скорость обучения: $lr = 0.1$.

4.4 Изменение learning rate

Для улучшения сходимости использовался **постепенно уменьшающийся learning rate**. Это позволяет модели более эффективно сходиться в процессе обучения, особенно в последних итерациях, когда прогресс в уменьшении ошибки замедляется.

В процессе обучения, если на текущей итерации средняя ошибка на тренировочных данных не изменилась более чем на 0.001 в относительном плане (по сравнению с предыдущей итерацией), то скорость обучения не меняется для следующей итерации:

$$\eta_{t+1} = \eta_t,$$

где: - η_t — learning rate на шаге t, - η_{t+1} — новый learning rate на шаге $t + 1$.

Если же изменение меньше заданного порога min_delta = 0.001, то

$$\eta_{t+1} = \eta_t \cdot \gamma,$$

где: - $\gamma = 0.5$ — коэффициент уменьшения learning rate.

Процесс обучения продолжается либо до 100 эпохи, либо при достижении сходимости. Скажем, что процесс обучения сошелся, если средняя ошибка на тренировочных данных была меньше min_delta в относительном плане (по сравнению с предыдущей итерацией) на протяжении 5 эпох даже с учетом изменения скорости обучения.

4.5 Предположения для проверки

Предполагается, что использование кастомного оптимизатора, учитывающего не только градиент, но и текущую структуру весов, улучшит сходимость по сравнению с использованием стандартного **SGD**.

4.6 Методика проверки

Для проверки гипотезы о лучшей сходимости кастомного оптимизатора была проведена серия экспериментов. В ходе эксперимента сравнивались результаты обучения с использованием стандартного **SGD** и кастомного оптимизатора. Мы будем анализировать следующие метрики:

- Ошибка восстановления на тестовой выборке при помощи метрик MSE и SSIM.
- Анализ графика изменения ошибки на тренировочных данных во времени.

Результаты будут приведены в следующем разделе.

5 Результаты

Для удобства дальнейшего повествования назовем новый оптимизатор CSGD (Custom SGD). В ходе эксперимента были проанализированы две модели автоэнкодера: одна с использованием стандартного стохастического градиентного спуска (SGD), а другая с CSGD, который включает дополнительный термин $A_t^\top A_t$ для модификации обновлений весов. Рассмотрим результаты на основе нескольких метрик: ошибки восстановления (MSE) и структурного сходства (SSIM).

5.1 Ошибка восстановления (MSE)

На графике, приведённом ниже (Рисунок 2), показано изменение ошибки восстановления на тренировочной выборке по эпохам для обеих моделей. Из графика видно, что CSGD (оранжевая линия) демонстрирует более быструю сходимость по сравнению с моделью, использующей стандартный SGD (синяя линия).

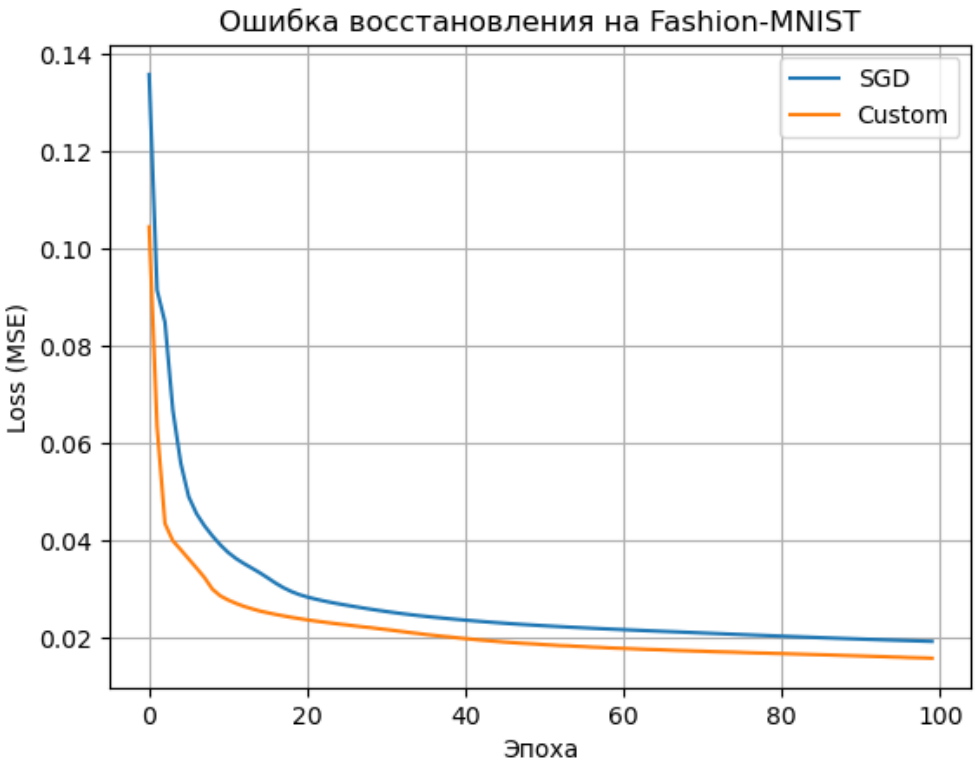


Рис. 2: Изменение ошибки восстановления на Fashion-MNIST

Как видно из графика, на более поздних этапах обучения, ошибка восстановления для обеих моделей стабилизируется, однако модель с CSGD сохраняет преимущества в ускорении сходимости.

5.2 Сравнение SSIM и MSE

В таблице ниже приведены значения метрик **SSIM** (структурное сходство) и **MSE** (среднеквадратичная ошибка) для обеих моделей на тестовой выборке.

Метрика	**SGD**	**Custom**
SSIM	0.6020	0.6555
MSE	15.0783	12.3370

Таблица 1: Сравнение SSIM и MSE для обеих моделей

Из результатов видно, что CSGD превосходит стандартный SGD по обеим метрикам. **SSIM** для кастомного оптимизатора выше, что указывает на лучшее восстановление структуры изображений. Также **MSE** для кастомного оптимизатора ниже, что подтверждает более точное восстановление изображений.

5.3 Итоги

Результаты показали, что использование CSGD позволяет улучшить как скорость сходимости, так и качество восстановления изображений, по сравнению с использованием стандартного стохастического градиентного спуска (SGD).

6 Заключение

В рамках данной курсовой работы было исследовано влияние использования кастомного стохастического градиентного спуска (CSGD), учитывающего структуру данных, на процесс обучения нейронных сетей, в частности на задаче восстановления изображений. Были рассмотрены две модели автоэнкодера: одна с использованием стандартного стохастического градиентного спуска (SGD), а другая с CSGD. Основное внимание уделено сравнению этих методов по метрикам **SSIM** (структурное сходство) и **MSE** (среднеквадратичная ошибка), а также по скорости сходимости.

Результаты показали, что CSGD превосходит стандартный SGD по обеим метрика. Также CSGD продемонстрировал более быструю сходимость на тренировочных данных по сравнению с классическим SGD.

Следовательно, можно сделать вывод, что использование CSGD имеет явные преимущества.

6.1 Планы на будущее

В дальнейшем планируется:

- Описать область применимости метода CSGD и провести дополнительные эксперименты для других типов нейронных сетей.
- Сформулировать теорему , утверждающую, что скорость сходимости метода CSGD лучше, чем у стандартного SGD, и для каких задач.
- Доказать теорему.

Эти исследования помогут более точно определить, в каких случаях CSGD может значительно улучшить результаты обучения, а также будут способствовать дальнейшему развитию теории об оптимизации нейронных сетей.

Дополнительные материалы, включая коды, данные и визуализации, доступны в репозитории на GitHub по следующей ссылке: `https://github.com/MiroshnichenkoNI/CSGD`

Список литературы

[1] D. P. Kingma, M. Welling, Auto-Encoding Variational Bayes, arXiv preprint arXiv:1312.6114, 2013.

[2] X. Ding, X. Zhang, N. Ma, J. Han, G. Ding, J. Sun, RepVGG: Making VGG-style ConvNets Great Again, Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 13733–13742, 2021.

[3] X. Zhang, Y. Wang, J. Huang, T. Guo, Y. Zhang, J. Sun, RepOptimizer: Residual Learning Meets Architecture Search for Training-Deploy Inconsistency, arXiv preprint arXiv:2205.15242, 2022.